AFRL-RX-WP-TR-2013-0210

COLLABORATIVE RESEARCH AND DEVELOPMENT (CR&D) III

Task Order 0090: Image Processing Framework: From Acquisition and Analysis to Archival Storage

Michael A. Jackson
BlueQuartz Software

MAY 2013
Final Report

Approved for public release; distribution unlimited.

See additional restrictions described on inside pages.

STINFO COPY

AIR FORCE RESEARCH LABORATORY
MATERIALS AND MANUFACTURING DIRECTORATE
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7750
AIR FORCE MATERIEL COMMAND
UNITED STATES AIR FORCE

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the USAF 88th Air Base Wing (88ABW) Public Affairs Office (PAO), and is available to the general public, including foreign nationals.

Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RX-WP-TR-2013-0210 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

_____//SIGNATURE//_____
CHRISTOPHER WOODWARD, Project Engineer
Metals Branch
Structural Materials Division

_____//SIGNATURE//_____
DANIEL EVANS, Chief
Metals Branch
Structural Materials Division

_____//SIGNATURE//_____
ROBERT T. MARSHALL, Deputy Chief
Structural Materials Division
Materials and Manufacturing Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
May 2013	Final	11 June 2010 – 18 April 2013

4. TITLE AND SUBTITLE
COLLABORATIVE RESEARCH AND DEVELOPMENT (CR&D) III
Task Order 0090: Image Processing Framework: From Acquisition and Analysis to Archival Storage

5a. CONTRACT NUMBER
FA8650-07-D-5800-0090

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62102F

6. AUTHOR(S)
Michael A. Jackson

5d. PROJECT NUMBER
4347

5e. TASK NUMBER

5f. WORK UNIT NUMBER
X0QU

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

By:
BlueQuartz Software
400 South Pioneer Blvd.
Springboro, OH 45066

For:
Universal Technology Corporation
1270 North Fairfield Rd.
Dayton, OH 45432

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Materials and Manufacturing Directorate
Wright-Patterson Air Force Base, OH 45433-7750
Air Force Materiel Command
United States Air Force

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/RXCM

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-RX-WP-TR-2013-0210

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES
Approved by 88ABW Public Affairs Office. Case Number: 88ABW-2013-4547; on 31 October 2013. Report contains color.

14. ABSTRACT (*Maximum 200 words*)
This research in support of the Air Force Research Laboratory Materials and Manufacturing Directorate was conducted from 11 June 2010 through 18 April 2013. The objective of this study was to develop a flexible and portable software frame work to manage evolving software for importing, manipulating and archiving microstructural images and data.

15. SUBJECT TERMS
DREAM3D, Finite Element Method (FEM), Integrated Computational Materials Science (ICME), Graphical Interface (GUI)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Christopher Woodward
Unclassified	Unclassified	Unclassified	SAR	49	19b. TELEPHONE NUMBER *(Include Area Code)* (937) 255-9816

TABLE OF CONTENTS

TABLE OF CONTENTS *(Cont'd)*

LIST OF FIGURES

LIST OF TABLES

1.0 SUMMARY

A software framework called **D**igital **R**epresentation **E**nvironment for **A**nalysis of **M**icrostructure (DREAM3D) has been created that allows a user to reconstruct three-dimensional materials data into a voxel representation where further statistical analysis can be applied to the data resulting in new insights into the internal structure of a material. In addition to the new software framework can generate synthetic microstructures that mimic a range of important material groups to the Air Force including poly-crystal metals, ceramics and composites. These synthetic structures can be used in microstructure simulations involving Finite Element Method algorithms, data acquisition sensitivity studies and linking material properties to material structure. The software is simple and easy to use due to its Graphical User Interface and allows the user to easily share their analysis with others using DREAM3D through a common file format. The software framework utilizes the HDF5 file format to store all input and processed data. HDF5 is an open-source file format that is designed for high performance and full data description so that every HDF5 file has a full description of the data stored in the file itself and can be queried as needed. Finally the DREAM3D software itself is open-source and anyone can download, use and extend DREAM3D by visiting the home website at dream3d.bluequartz.net.

2.0 INTRODUCTION

In the last four years there has been a large push in the materials science community to utilize computational materials science in more and more areas. This push is referred to ICME or Integrated Computational Materials Science and this paradigm shift is starting to take hold in the various laboratories worldwide. But with every new shift comes issues that the originators had never considered and one that stands out is the "Integrated" in ICME. From the originators of ICME's ideas was that all the computational codes could communicate between each other easily and thus the "Integrated" would be satisfied. Unfortunately in the research world new computer codes rarely are able to communicate with other codes due to differences in input and output file formats, computer architectures or communications protocols. Codes that are initially created during a single research grant period such as a contract or a PhD dissertation typically are a "proof-of-concept" code base that can only read a single set of inputs and are not designed with robustness or flexibility in mind. To make matters even worse the documentation is lacking and the human-computer interface is substandard at best. All of these combine to ensure that the codes are never able to be used easily by other researchers who may in fact be interested in the algorithms being used. What is needed is a software framework that acts as an umbrella that allows all of these disparate codes to communicate seamlessly with each other so that new any new algorithms can build upon algorithms that have been proven in both research and commercial settings. This same software umbrella framework would allow the various algorithms to be combined into series of operations collectively referred to as a pipeline where the operations are know as filters and each filter is aware of a common data structure and can thus communicate, or talk, to the other algorithms. The new software framework would be easily accessible to any researcher who wanted to simply use the algorithms for their own research or add to the software framework new algorithms that they want to share with the rest of the community. To this end the current contract's answer to this problem is the DREAM3D software environment.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

3.1 DREAM3D

DREAM3D stands for the **D**igital **R**epresentation **E**nvironment for **A**nalysis of **M**icrostructure in **3D** and is the culmination of 3 years of development work to integrate a wide range of individual software tools into a single cohesive software package that is usable by the every-day materials scientist. DREAM3D's main capabilities are the reconstruction of EBSD (Electron BackScatter Diffraction) data into a 3 dimensional representation that can simply be visualized for a greater understanding of the material or analyzed with a range of statistical methods to quantify the microstructure in order to link material properties to a specific type of microstructure. Another capability of DREAM3D is to generate a purely synthetic microstructure based on statistics. Those statistics can either be derived from real data or created through an ancillary program called "**StatsGenerator.**" Once these 3D structures, or models, are created using DREAM3D the user can now export the model into a data format suitable for use in any number of Finite Element Methods (FEM) simulations for such things as shocked materials, simulated heat stress experiments or simple mechanical loading simulations. Each type of analysis gives the materials scientist a deeper understanding of the link between a material's properties and its microstructure.

3.2 Software Development Practices

During the development of DREAM3D best practices for computer software development were employed where possible. These best practices are industry standard ideas that have been developed and refined over the last 20 years in the software engineering domain. Practices such as having a central location to store the source codes, a system to control and track each revision to the software, creating small targeted programs called "Unit Tests" that allow changes to the software to be checked against known standards and using accepted documentation practices were all used during the development of DREAM3D. Once a stable version of DREAM3D has occurred precompiled binaries for Microsoft Windows, Apple OS X and Linux were posted to the DREAM3D website located at http://dream3d.bluequartz.net and hosted by the contractor. In addition a collection of interested users was collected and these users were sent brief messages informing them of the new release.

3.2.1 Source Code Repository

During the development of computer software where multiple programmers are involved tools that allow each engineer to rapidly make changes to the project are invaluable. For this reason a source code repository was created where each programmer can both get the latest versions of the source code files and deposit their own changes. The particular protocol decided upon by the DREAM3d team was "Git" as it has a good balance of ease of use, cross platform use and powerful tracking abilities. Programmers were encouraged to "push" their code to the repository where it is "merged" with everyone else's changes. When the programmer needs an up-to-date version they can simply "pull" the changes from the central Internet based server onto their local workstations. This workflow where one pulls, merges and then pushes is a standardized workflow in today's software engineering domain and enables those writing software tools to easily collaborate with other programmers around the world. In order to make the software codes developed under this contract easily accessible a server was provisioned that serves as a central location to store the source code repository. Registered developers can access the

repositories for reading and writing through the Secure Shell (SSH) protocol. The source code repository is in this respect 'private' to just the DREAM3D developers and anyone else that meets the proper criteria and has asked for access to the repository.

3.2.2 External Libraries Needed for DREAM3D

In the spirit of reusing codes that are already in existence and to speed the development cycle, DREAM3D leverages several open source libraries that contain reusable components that allow DREAM3D to be deployed on all the industry standard desktop class operating systems and also allow a more rapid development and revision cycle.

3.2.3 CMake

CMake is used to describe how the various software components are compiled together and is able to generate configuration files for a large number of popular development environments including Visual Studio for Microsoft Windows, Xcode for Apple's OS X and Makefiles for typical Linux installations.

3.2.4 Boost

Boost is an open-source software library that has many reusable C++ components. In this instance we are using Boost for low-level C++ object memory management and error detection.

3.2.5 Intel Threading Building Blocks (TBB)

TBB is used by software developers to create versions of their codes that can be executed in parallel on multiple CPUs or CPU cores. The TBB library is open-source and is mainly backed by the Intel Corporation and is quickly becoming a standard in the software-programming domain.

3.2.6 Qt

The Qt software framework is an open-source software development framework that has successfully abstracted the need to write platform specific software code. This allows a single developer to target the three main operating systems in use today without any major code changes for each platform.

3.2.7 Qwt

Qwt is a 2D plotting software library that uses the Qt framework as its drawing canvas making the use of Qwt a natural fit for applications that present a GUI created with the Qt software framework.

3.2.8 Eigen

Eigen is a Matrix and Array manipulation and linear algebra software package. Because the code is based on C++ structures and not the typical FORTRAN language this allows Eigen to be more easily used on cross platform projects.

3.2.9 DOxygen

Doxygen is used to document source code by enumerating code comments with special markers that the Doxygen parser can find easily. The programmer has the ability to select what types of output to be produced from LaTeX, HTML or Rich Text Format (RTF). Doxygen was used to document as much of the source and filters as possible.

3.2.10 System Requirements

All the applications developed under this contract work on the vast majority of desktop computers including computers designated as "Workstations." The limiting factor for running the application is that the computer has enough memory available to process the input data.

Table 1. General System Requirements

Operating System	
Apple OS X	OS X versions 10.6.8, 64 Bit, Intel Arch
Microsoft Windows	XP(sp3), Vista, 7, 8, 32/64 Bit, Intel Arch
Linux	Red Hat Linux Version 6, 64 Bit, Intel Arch

4.0 RESULTS AND DISCUSSIONS

In the current incarnation DREAM3D focuses on the processing of orientation data collected from various manufacturers instruments and generation of synthetic microstructures. This focus was a design decision so that the programmers could concentrate on creating solid algorithms for processing data instead of trying to reinvent a 3D viewer. For visualization duties the user is encouraged to utilize the free ParaView software available from http://www.paraview.org. At current count there are approximately 118 processing filters that are included in DREAM3D that gives the user a wide range of processing choices and fall into the following general categories.

- Reconstruction of Orientation Data
- Calculating Statistics on the reconstructed structure
- Generating a synthetic microstructure
- Generating a surface mesh suitable for use in simulation codes

4.1 Data Representation

DREAM3D uses a data structure that is based on the concepts of combinatorial topology and common methods for describing mesh structures. Any topological network (in 2- or 3-D) can be described with a the following hierarchy:

If the mesh (or network) is 2-D, then the volume element does not exist and the face element is the highest-level element. For typical voxel-based data, the voxels are the Volume elements and are referred to within DREAM3D as cells. Similarly, in a surface mesh with triangular patches, the triangles are the face elements and are referred to within DREAM3D as faces. This topology is required to describe the "mesh" or "structure" of the data itself and is not/cannot be adjusted/defined by the user.

However, once the topology of the "structure" is set, the user can begin grouping topological elements into higher-level features of the "structure." Below is an example of this grouping for a voxel-based dataset of a polycrystalline metal.

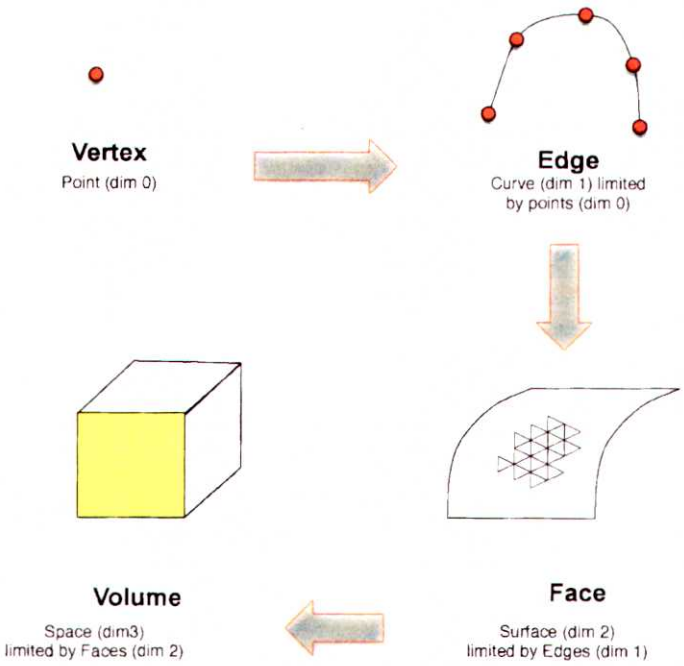

Figure 1. DREAM3D Hierarchy of Topology

DREAM3D uses two additional levels above the topological levels required to define the "structure" of the data. These two levels are called: field and ensemble. These levels allow the user to group topological elements together based on criteria of similarity. For the example above, the cells are grouped to identify fields (i.e. grains) by applying a criterion of similar crystallographic orientation (i.e. neighboring cells with similar orientation are said to belong to the same field). The fields are then grouped to identify ensembles by a criterion of similar phase (i.e. all grains of the same phase in the dataset are said to belong to the same ensemble). The grouping criteria are at the discretion of the user, because these additional levels are not required for description of the "structure," but rather are organizational levels for describing the information that lives on the "structure."

At each level, DREAM3D creates a map to store information/data about the individual elements at that level. Additional information about the maps of the DREAM3D data structure and "typical" data are given below:

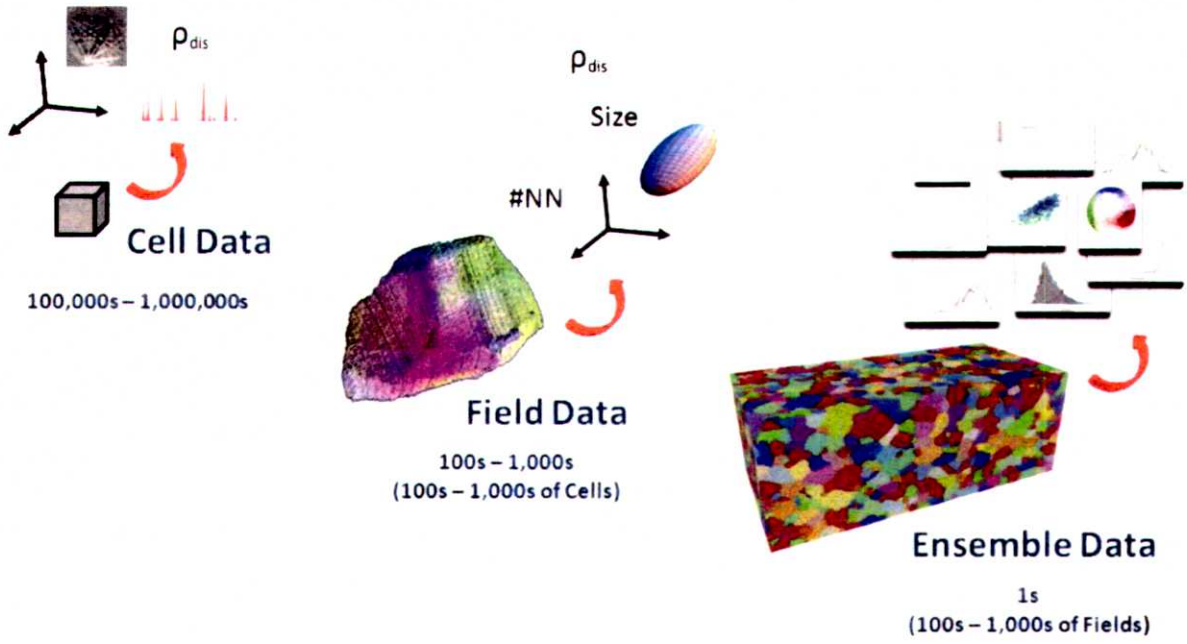

Figure 2. DREAM3D Internal Data Structures and Organization

4.1.1 Vertex

- Map of attributes associated with single points.
- Measured values (i.e. Orientation, Chemistry, Gray scale, etc.) - generally measured data is not represented at the vertex level, but the user could choose to store measured values at the vertices rather than the more typical cell-centered convention.
- Calculated values (i.e. Curvatures, Coordination, etc.) - these calculated values are typically values associated with the connectivity or geometry of the "mesh" or "structure" itself.

4.1.2 Edge

- Map of attributes associated with edges of faces.
- Measured values - data is not typically measured at the edge level.
- Calculated values (i.e. Curvatures, Coordination, etc.) - these calculated values are typically values associated with the connectivity or geometry of the "mesh" or "structure" itself.

4.1.3 Face

- Map of attributes associated with surface patches.
- Measured values - data is not typically measured at the face level.
- Calculated values (i.e. Bounding Field Ids, Normals, Curvature, etc.) - these calculated values are typically IDs to the higher level maps (field and ensemble), related to gradients of the measured values, values associated with the connectivity or geometry of the "mesh" or "structure" itself or relationships to values calculated in the higher level maps.

4.1.4 Cell

- Map of attributes associated with single data points - often these data points are not truly volumes, but rather point-probe measurements that are homogenized over the volume nearest to each probe-point.
- Measured values (i.e. Orientation, Chemistry, Gray scale, etc.) - typically this is the level at which most data is actually acquired.
- Calculated values (i.e. Field Ids, Kernel Avg. Misorientation, Euclidean Distance, etc.) - these calculated values are typically IDs to the higher level maps (field and ensemble), related to gradients of the measured values or relationships to values calculated in the higher level maps.

4.1.5 Field

- Map of attributes associated with sets of data points - generally elements in this map are sets of contiguous cells that have some aspect of similarity, often in their measured values (i.e. orientation, chemistry, gray scale, etc.).
- Measured values (i.e. Avg. Stresses, Avg. Strains, etc.) - some measurement techniques are capable of measuring values averaged over an entire field (i.e. High Energy Diffraction Microscopy).
- Calculated values (i.e. Avg. Orientation, Size, Shape, No. Of Neighbors, etc.) - Generally there is little measured data at this level and these values are calculated from averaging the measured data from the cells that constitute each field or geometrically describing the size and/or shape of the set of cells.

4.1.6 Ensemble

- Map of attributes associated with sets of fields.
- Measured/Set values (i.e. Crystal Structure, Phase Type, etc.) - measured values at the ensemble level are usually user-defined parameters during the data collection or during analysis within DREAM3D.
- Calculated values (i.e. Size Distribution, No. Of Fields, ODF, etc.) - These values are typically descriptions of the distribution of values from the lower level field map.

4.2 DREAM3D Data Outputs

DREAM3D is capable of creating files for many different uses including analysis, visualization and archiving. During any DREAM3D session the user can select to save their processed data to an HDF5 based file format that will store all data that is currently resident in memory. HDF5 was built for extremely large data sets to archive data generated on High Performance Computing (HPC) systems. HDF5 is open-source ensuring that data generated is always available to the user. If the user wishes to plot various statistics DREAM3D can output compatible tabular types of files that modern statistical analysis and plotting packages are capable of reading. And finally if the user wishes to visualize the original and/or processed data DREAM3D can create files for products such as the free ParaView or the commercial Avizo Fire data visualization programs.

4.3 Extending DREAM3D

As more features have been added to DREAM3D experts in the domain of 3D materials science have started discussions regarding how to integrate their own computer codes into DREAM3D. Being conscious that not every organization is in a position to release their intellectual property, a novel plugin system has also been incorporated into DREAM3D that would allow an organization to build on the core of DREAM3D with their own proprietary software models, simulations and filters. For those scientists that have the necessary programming skills DREAM3D can be extended through the direct addition of filters into the core of DREAM3D or by creating their own collection of filters that are grouped into an extra library called a plugin. These options can cover a wide range of possibilities for organizations wanting to extend DREAM3D and due to the use of a non-viral open-source license for DREAM3D this is entirely possible. The following are just two possible scenarios that the plugin system creates.

4.3.1 Proprietary Algorithms

One possible scenario is that a group would like to leverage DREAM3D for its base set of filters but add their own processing algorithms into DREAM3D but not have to release either the source code for the algorithms or the actual binary plugin itself. By using DREAM3D's plugin system this is entirely possible as the group can gain access to the complete source code for DREAM3D and compile and distribute within their organization their own custom version of DREAM3D.

4.3.2 Monetizing DREAM3D

Another scenario for extending DREAM3D would see custom processing filters being created as plugins for DREAM3D and sold through various means to those organizations needing the seller's algorithms.

5.0 CONCLUSIONS

Over the last few years as DREAM3D matured into an easily usable application. It has gained a foothold in the 3D microstructure reconstruction and modeling domains and is poised to become a driving force in the materials community by offering quality data analysis and models at a tremendous low cost to the user. Establishing this early adopter user base by influential researchers in the field is an important result of this work. Their recommendations concerning which computational tools to use reverberate through the industry and help to establish DREAM3D as a viable research and development platform for many in the materials science community.

5.1 Accelerating Research in 3D Materials Science

By offering a free to use, quality software package for the reconstruction and analysis of 3D materials data DREAM3D can accelerate research in those areas typically slowed down by custom software solutions. For example in the past when a researcher was given a 3D EBSD data set they would have to manipulate the files there were given into more efficient data structures through custom written software. Writing and debugging this software can steal time away from the actual research that needs to be performed. By using DREAM3D for these basic reconstructions and manipulations the research can quickly transform the data set into something immediately useful within minutes instead of having to wait months for a custom and often non-robust software tool to be written. This compression of time will in effect accelerate their research as they now have more time available for the actual research instead of writing file parsers, alignment routines and segmentation routines. Table 2 details a small portion of the known users of DREAM3D from around the world.

Table 2. Current Users of DREAM3D

Air Force Research Laboratory	Army Research Laboratory
Carnegie Mellon University	Purdue University
Los Alamos National Lab	Lawrence Livermore National Lab
Naval Research Labs	Argonne National Lab
GE Global Research	Sandia National Lab
Université de Lorraine, Metz France	University of California, Santa Barbara
Idaho National Lab	University of Manchester
NASA	Cornell University
University of Michigan	King Abdulla University of Technology

In addition to the wide range of users worldwide the internet home page for DREAM3D averages about 600 unique visitors per month and the software has been downloaded more than 2000 times.

5.2 Presentations and Workshops

The following is a list of presentations and workshops that the contractor has directly been involved in creating and presenting on the DREAM3D software.

2010 MRSEC Workshop on 3D Materials Science, Carnegie Mellon University, Pittsburg Pa

2011 MRSEC Workshop on 3D Materials Science, Carnegie Mellon University, Pittsburg Pa

2013 MRSEC Workshop on 3D Materials Science, Carnegie Mellon University, Pittsburg Pa

2011 TMS Annual Meeting, San Diego Ca, Using DREAM3D

2012 1rst Annual TMS Conference on 3D Materials Science

2013 TMS Annual Meeting, San Antonio Tx. Overview of DREAM3D's capabilities

2013 2 Day DREAM3D workshop for AFRL participants

6.0 RECOMMENDATIONS

While DREAM3D has a solid core of algorithms to deal with digital microstructures, that initial data will most likely have come from an EBSD data source or data source that generates orientation angle data types. Materials scientists often utilize other data modalities such as direct optical imaging of the microstructure where the "data" is actually a picture or image. This image data is usually stored in some industry standard format such as Tif or PNG. Currently DREAM3D does not have the processing algorithms to handle this image data well and the developers are looking into the most efficient way to be able to process this type of data. One possible solution would be to introduce another library that does the image processing already and simply have DREAM3D act as a data staging level and then pass the image data to this new library. Ideally the library would be open-source with a compatible open-source license. There are a number of candidates that were investigated late in the development of DREAM3D which were the "Image Processing Toolkit" (ITK) and the "Open Computer Vision" (OpenCV) software library. Both have a large number of image processing routines that could be made available to DREAM3D users.

Another issue that needs to be addressed is the long-term maintenance of DREAM3D. The source codes are all open-source but a more suitable neutral hosting location needs to be found so that any organization can download the codes if they wish to use or expand them. Having this truly open-source location where anyone can get the sources without first having to go through the current developers for access should help further accelerate the development and usage of DREAM3D. The current developers also receive help requests and feature requests through the DREAM3D web site which will need to transition to a more crowd friendly solution such as a public mailing list or group forum hosted on the Internet. All of these items add to the maturity and user base of DREAM3D ensure its continued development and success.

7.0 REFERENCES

[1] Cerrone, Tucker, Stein, Rollet, & Ingraffea (2012). "Micromechanical Modeling of René88DT: From Characterization to Simulation." 2012 Joint Conference of the Engineering Mechanics Institute and 11th ASCE Joint Specialty Conference on Probabilistic Mechanics and Structural Reliability (EMI/PMC 2012) June 17-20, 2012.

[2] Groeber, M., Ghosh, S., Uchic, M.D., and Dimiduk, D.M. (2008a). "A framework for automated analysis and simulation of 3D polycrystalline microstructures. Part 1: Statistical characterization." *Acta Materialia*, 56(6), 1257-1273.

[3] Groeber, M., Ghosh, S., Uchic, M.D., and Dimiduk, D.M. (2008b). "A framework for automated analysis and simulation of 3D polycrystalline microstructures. Part 2: Synthetic structure generation." *Acta Materialia*, 56(6), 1274-1287.

[4] Tucker, J.C., Chan, L.H., Rohrer, G.S., Groeber, M.A., and Rollett, A.D. (2012). "Comparison of grain size distributions in a Ni-based superalloy in three and two dimensions using the Saltykov method." *Scripta Materialia*, Acta Materialia Inc., 66(8), 554-557.

APPENDIX A: DREAM3D USER MANUAL

Overview of the User Interface

As shown in Figure A-1 DREAM3D has 4 main areas of its user interface:

1. The Filter Library
2. The Filter List
3. The Errors Tab
4. The Pipeline Area

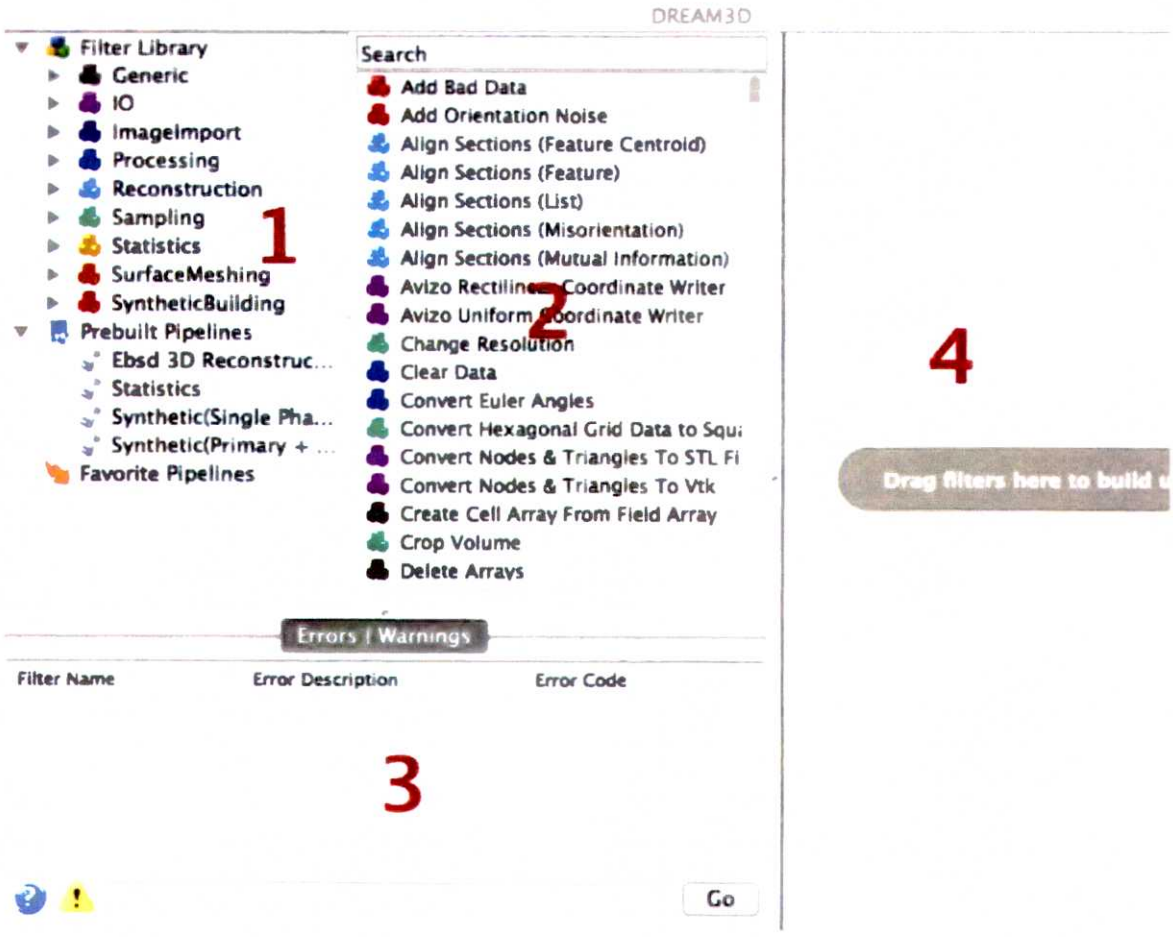

Figure A-1. Four Main Areas of DREAM3D User Interface

The Filter Library

This area of the user interface is broken into 3 basic groups: _Filter Library_, _Prebuilt Pipelines_ and _Favorite Pipelines_. The _Filter Library_ section contains the various groups

that the filters are organized into. The user can click a specific group and just the filters associated with that group will be displayed in the _Filter List_ area of the user interface.

Prebuilt Pipelines

DREAM3D provides several _Prebuilt Pipelines_ that can aid the new user in getting started with DREAM3D. Simply double clicking a preset will clear any current filters in the pipeline area and populate the pipeline area with the filters from the Pipeline Preset.

Favorite Pipelines

After the user builds a pipeline that they may want to save for later the user can use the _Pipeline_ menu and select the "Add Favorite" menu to save that specific pipeline configuration. If at a future point in time the user wants to remove the favorite from the _Favorites_ list they user can select from the "Add Favorite" menu and choose *Remove Favorite* to remove it from the list.

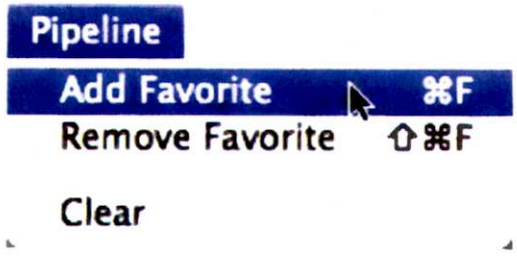

Figure A-2. Pipeline Menu

Filter List

This section lists the filters that are associated with a specific group that is selected in the _Filter Library_ area. If the _Filter Library_ is itself selected then all filters will be shown in this list. The user typically will select a filter from this list and drag it over to the pipeline area.

Errors & Warnings Tab

This area displays any errors and/or warnings associated with the filter if it is actively being used in a pipeline. The display of the errors can be toggled on and off by pressing the appropriate buttons. Also clicking the "?" icon will display all of the help (including filter reference, tutorials and user manual) in the users default web browser.

Creating a Pipeline

In DREAM3D the user processes their data by creating what is known as a _Pipeline_ which is constructed out of a series of _Filters_. By chaining together a series of filters the data is processed in quick succession and can ultimately be saved to a number of different file formats. The user should be aware that all processing on the data is done in place, i.e., there is only a single copy of the data resident in memory at any one time.

Building a Pipeline

In order the build a pipeline the user can either double click on a particular filter or drag the filter from the *Filter List* into the *Pipeline Area*.

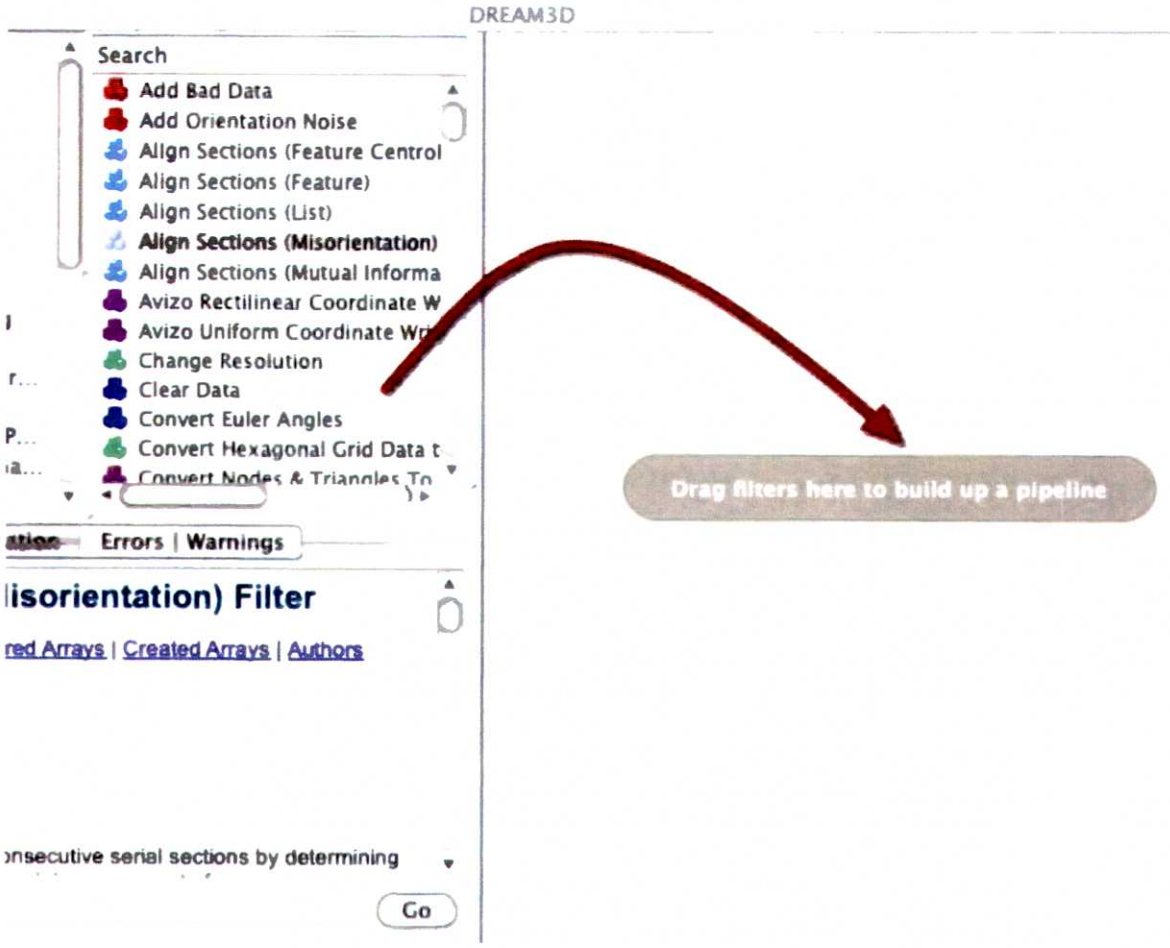

Figure A-3. Empty Pipeline Area

The user can place multiple filters into the pipeline to generate a workflow of data input, analysis and data output filters. The user can reorder the filters by simply using drag-and-drop of a selected filter in the pipeline and moving it into a new position in the pipeline. As filters are placed into the pipeline a basic error checking algorithm called the **Preflight** is executed on the entire pipeline. During the preflight stage the pipeline will have each filter check its input parameters and the availability of required data arrays. Figure A-4 shows a pipeline ready to execute with no errors.

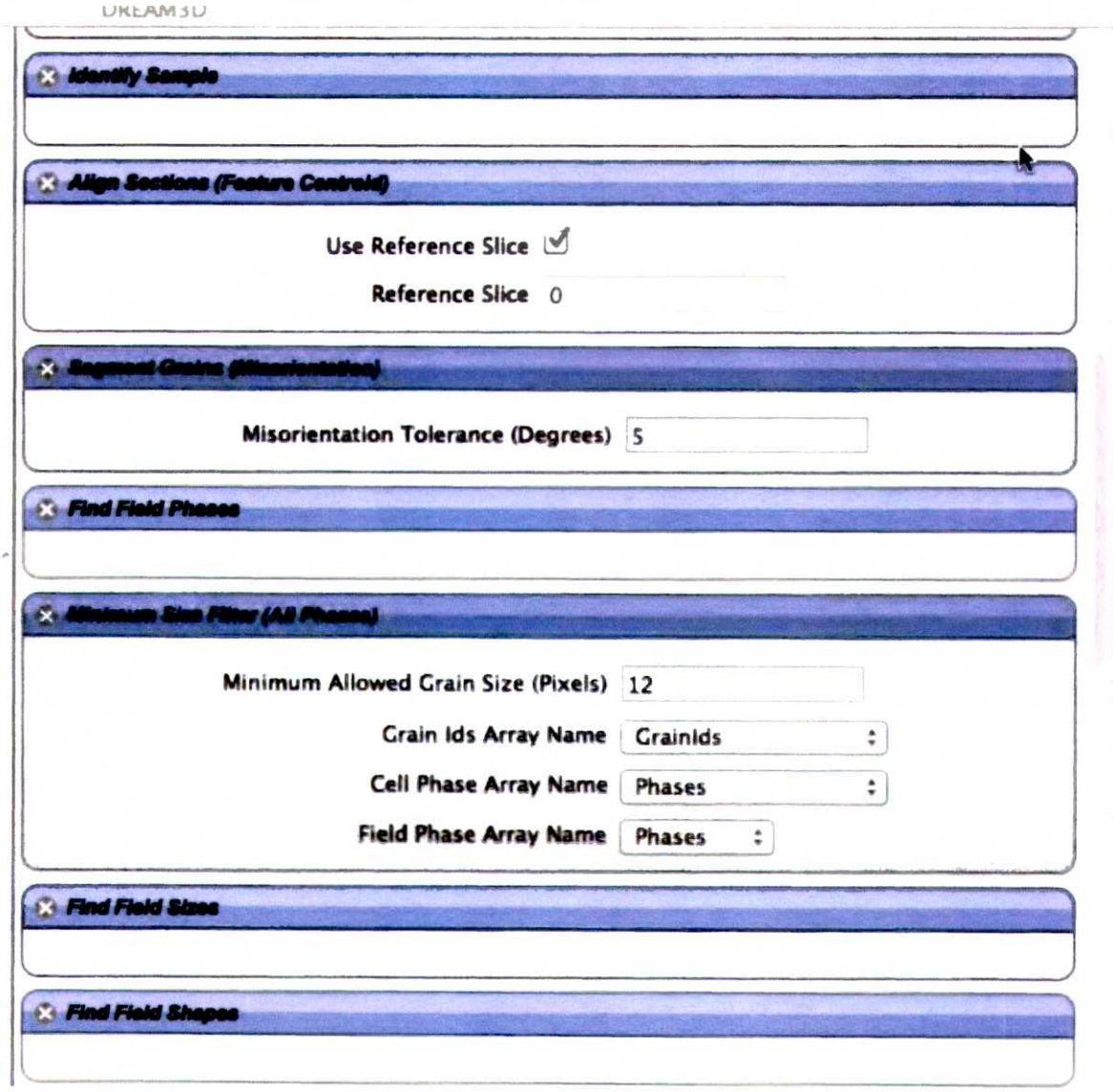

Figure A-4. Correctly Populated Pipeline

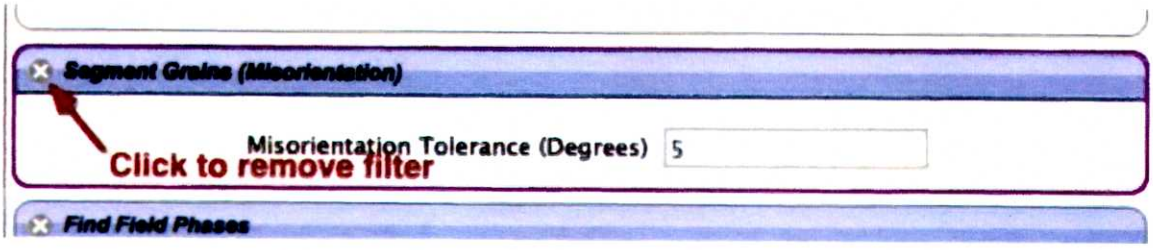

Figure A-5. Selected Filter in the Pipeline & How to Remove the Filter

To remove a filter from the pipeline the user can simply use the mouse to click the small **"X"** (Figure A-5) icon in the upper left corner of the filter.

If the preflight did not complete successfully the user will see the offending filters show a red outline and red title area. This indicates that one or more of the input parameters have an error or the filter requires data that will not be available at that point in the pipeline. By looking at the error table the user can read the error message from the pipeline and act accordingly. An example pipeline with errors is shown in Figure A-6.

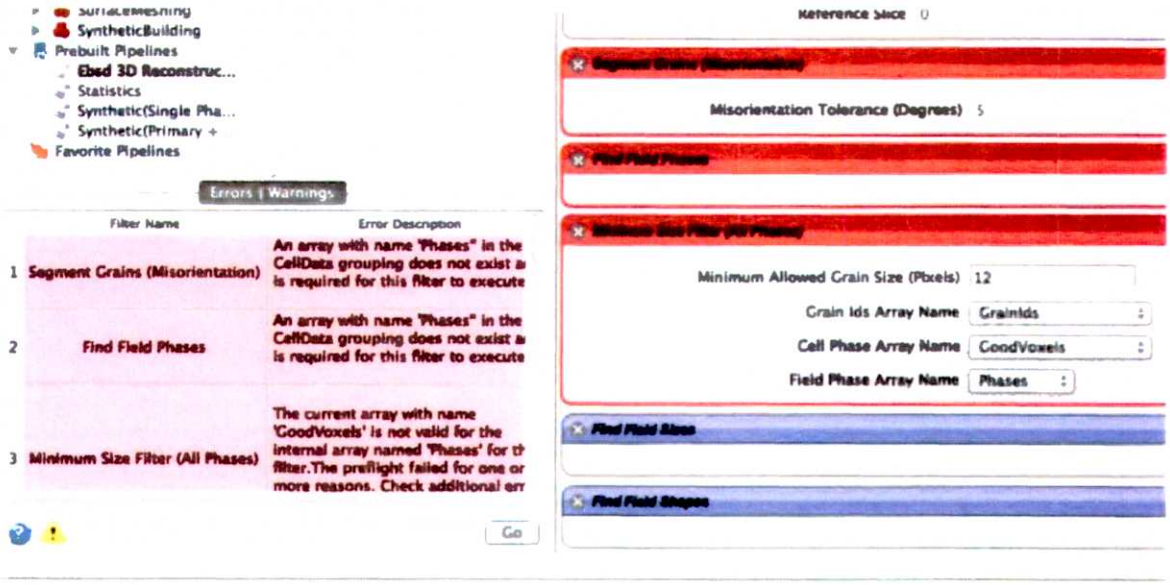

Figure A-6. Errors in the Constructed Pipeline

Once the errors are resolved the user can now execute the pipeline and generate the outputs. Because the DREAM3D native file format stores the complete state of the data arrays the user can use this idea to add *checkpoints* to the pipeline in the case of a long running pipeline.

Figure A-7. Errors & Warnings Area

Pipeline Area

This area is where the user will construct their pipeline by either double clicking on a filter in the filter list area or dragging a filter from the filter list and dropping the filter into the pipeline area. Filters in this section can be rearranged by simply dragging the filter into a new location. As the user updates the pipeline a process called the **Preflight** will be executed to make sure that the pipeline will have all the necessary data available during the actual execution of the processing pipeline.

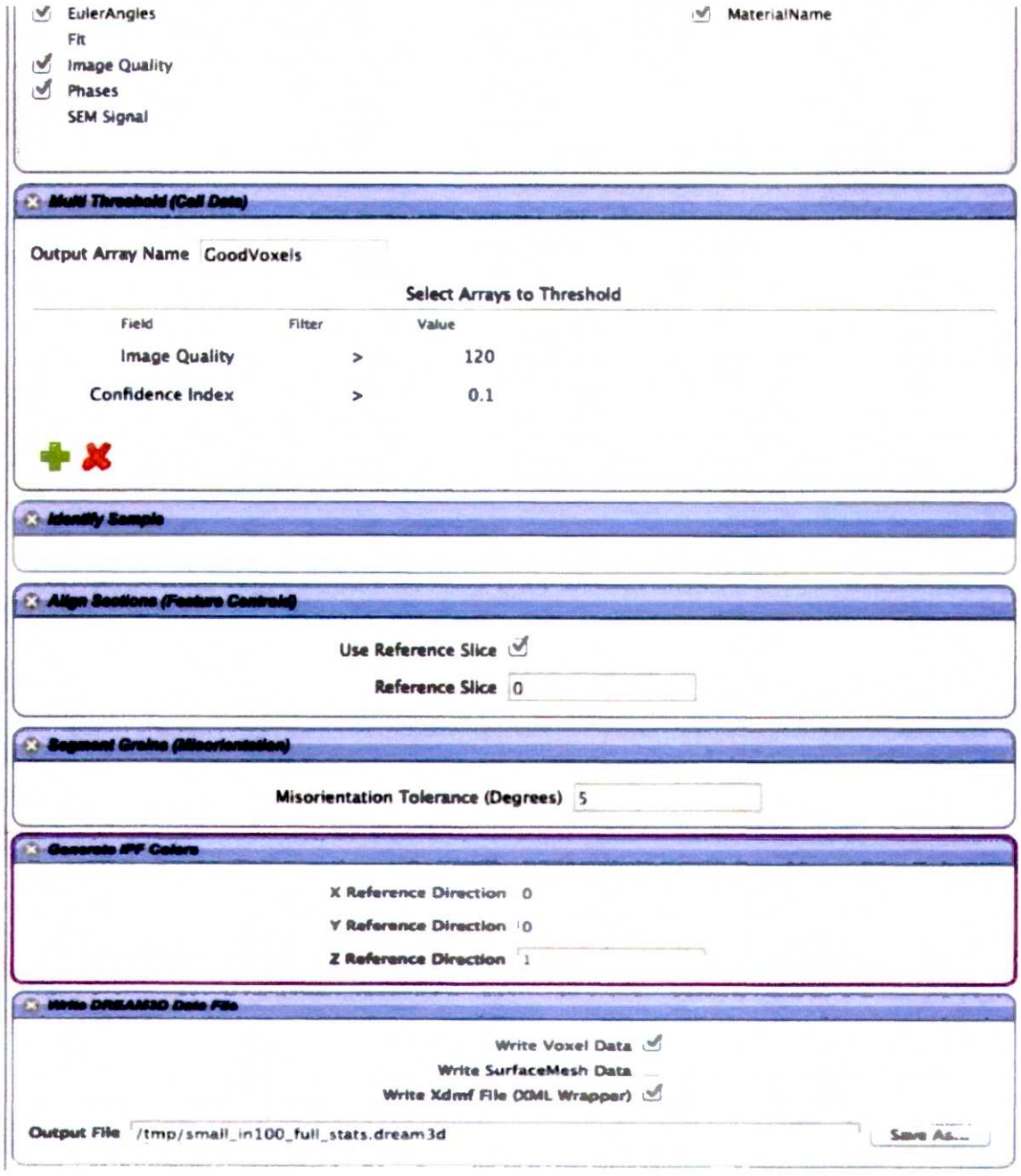

Figure A-8. Pipeline Area Populated with Filters

Import, Export & Favorite Pipelines

DREAM3D has the feature that will allow the user to save their pipelines to an external text file on the file system of the computer they are currently using. This facilitates easy exchange of pipelines between collaborators.

Saving and Loading a Predefined Pipeline

DREAM3D allows the user to Export and Import pipelines via the **File** menu and the **Save Pipeline** and **Open Pipeline** menu items. The pipeline files are saved in the *.ini* file format popular among many other programs. This file format is a simple text file with delineated sections with *Key::Value* entries. This makes the sharing of Pipelines mostly transparent between users. If a pipeline has filters that require files to be read from or written to and the pipeline file is passed to another user that does not have those same paths then the new user may have to make the adjustments inside of DREAM3D after the pipeline is loaded.

Figure A-9. File Menu

The following is a partial listing of a 3D EBSD reconstruction pipeline saved as a pipeline file:

```
[PipelineBuilder]
Number_Filters=15

[0]
Filter_Name=ReadH5Ebsd
InputFile=/Users/Shared/Data/Ang_Data/Small_IN100_Output/Small_IN100.h5ebsd
ZStartIndex=1
ZEndIndex=117
UseTransformations=false
ArraySelections_VoxelCell\size=4
ArraySelections_VoxelCell\1\VoxelCell=Confidence Index
ArraySelections_VoxelCell\2\VoxelCell=EulerAngles
ArraySelections_VoxelCell\3\VoxelCell=Image Quality
ArraySelections_VoxelCell\4\VoxelCell=Phases
ArraySelections_VoxelField\size=0
ArraySelections_VoxelEnsemble\size=2
ArraySelections_VoxelEnsemble\1\VoxelEnsemble=CrystalStructures
ArraySelections_VoxelEnsemble\2\VoxelEnsemble=MaterialName
```

ArraySelections_SurfaceMeshPoint\size=0

ArraySelections_SurfaceMeshFace\size=0

ArraySelections_SurfaceMeshEdge\size=0

ArraySelections_SolidMeshPoint\size=0

ArraySelections_SolidMeshFace\size=0

ArraySelections_SolidMeshEnsemble\size=0

[1]

Filter_Name=MultiThresholdCells

OutputArrayName=GoodVoxels

ComparisonInputs\size=2

ComparisonInputs\1\ArrayName=Image Quality

ComparisonInputs\1\CompOperator=1

ComparisonInputs\1\CompValue=120

ComparisonInputs\2\ArrayName=Confidence Index

ComparisonInputs\2\CompOperator=1

ComparisonInputs\2\CompValue=0.100000001490116

[2]

Filter_Name=IdentifySample

[3]

Filter_Name=AlignSectionsFeatureCentroid

UseReferenceSlice=true

ReferenceSlice=0

Creating a Favorite Pipeline

If the user creates a pipeline that they intend to use several times then they may want to save the pipeline to the *Favorites* area. Using the **Pipeline** menu and selecting the **Add Favorite** can accomplish adding the current pipeline to the Favorites list. If the user wants to remove a favorite at any time they simply need to select the **Remove Favorite** menu item.

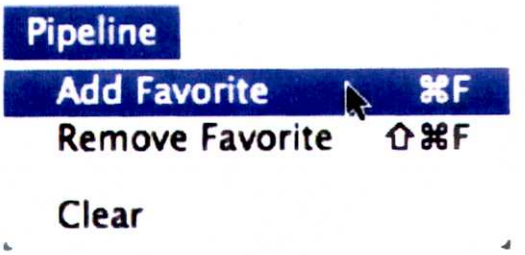

Figure A-10. Pipeline Menu

Clearing the Pipeline

As a convenience for those situations where the user has created a large pipeline and they would like to delete every filter from the pipeline the **Pipeline Menu** also has a menu item to remove every filter from a pipeline and an associated keyboard short cut.

Native DREAM3D File Format

HDF5

DREAM3D uses the [HDF5](http://www.hdfgroup.org) as its native file format. HDF5 stores data in a hierarchical format with complete descriptors for the data stored in the file. HDF5 is open-source and a number of tools exist that allow a user to view and manipulate HDF5 data files. Once such tool is the free HDFView (http://www.hdfgroup.org/hdf-java-html/hdfview/index.html#download_hdfview) from "The HDF Group" which is a java-based program that can view and edit HDF5 data files.

Each array is stored as an individual data set in HDF5 under one of several different types of *DataContainer* storage groups.

- Voxel DataContainer
- SurfaceMesh DataContainer
- SolidMesh DataContainer

Depending on the type of data created the actual data array that a user may be interested in may be stored in various subgroups in the HDF5 file. For example with a Voxel Data Container there are three types of Data: Cell, Field and Ensemble. Arrays created for each type are stored in subgroups. For example with Voxel based data the following HDF5 group organization is used:

- VoxelDataContainer (_Group_)
  - DIMENSIONS (_Dataset_)
  - ORIGIN (_Dataset_)
  - SPACING (_Dataset_)
  - CELL_DATA (_Group_)
    - GrainIds (_Dataset_)
    - EulerAngles (_Dataset_)
    - IPFColors (_Dataset_)
  - FIELD_DATA (_Group_)
    - Phases (_Dataset_)

        - AvgEulerAngle (_Dataset_)
    - ENSEMBLE_DATA (_Group_)
        - CrystalStructures (_Dataset_)
        - MaterialName (_Dataset_)

In the above example we have shown a number of data arrays for each group type. The user should note that the data sets shown in the above example are hypothetical and may or may not appear in data sets that the user generates.

- SurfaceMeshDataContainer (_Group_)
    - Faces (_Dataset_)
    - Vertices (_Dataset_)
    - CELL_DATA (_Group_)
        - Normals (_Dataset_)
        - Centroids (_Dataset_)
        - Curvature (_Dataset_)
        - IPFColor (_Dataset_)
    - POINT_DATA (_Group_)
        - VertexNormal (_Dataset_)

In the above we show how a surface mesh is stored internally in the native DREAM3D data file.

NOTE: DREAM3D is perfectly capable of storing **both** the voxel data and the surface mesh data in the same file if the user so desires. This allows all the data to stay together and more easily shared among collaborators.

Appendix B: H5EBSD Data File Specification

Introduction

The EBSD Data from multiple vendors is stored in a data file using the HDF5 file format library. While the general layout of the HDF5 file is the same between vendors there are details that are NOT the same between vendors because each vendors chooses to save different types of data. The top-level datasets that deal with the basic volume information is the same for every file.

Orientations, Reference Frames and Coordinate Systems

DREAM3D's origin follows the sample coordinate system so that the physical location of the 0 row and 0 column voxel should visually appear in the lower left corner of a computer graphics display. Sample Coordinate System (White) overlaid with EBSD Coordinate System (Yellow).

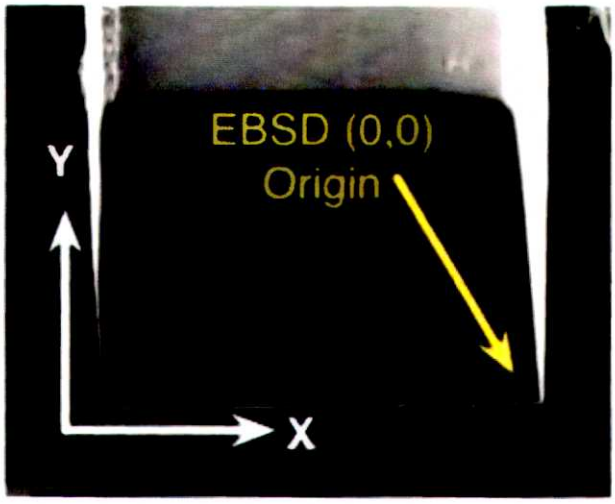

Figure B-1. TSL Coordinate System

Commercial EBSD acquisition systems do not typically follow this convention and DREAM3D needs input from the user so that the proper transformations to the data can be applied during the Reconstruction and other analysis. Commercial EBSD software packages allow for some initial transformations of the data in which case the DREAM3Denvironment does not have any way of determining if those transformations have already occurred. During the import process the user is asked a few questions regarding the orientation of their EBSD data in relation to the sample coordinate system. Currently there are 3 items that must be answered otherwise undefined behavior may occur during any of the analysis algorithms.

- Some Commercial EBSD acquisition systems allow for a **scan rotation**, which rotates the sample coordinate system, 180 degrees around the Z-axis.
- Should we rotate the Euler angles to bring them in line with the sample reference frame?

HDF5 File Layout

Table B-1. H5EBSD Root Node Specification

H5Ebsd Specification		
Root Level Datasets and Group	Attribute Name "FileVersion" Current Value="4"	
Name	HDF5 Type	Value
Index	H5T_NATIVE_INT32	List of all Slice index values
EulerTransformationAngle	H5T_NATIVE_FLOAT	
EulerTransformationAngle	H5T_NATIVE_FLOAT	
Manufacturer	H5T_STRING	The **Manufacturer** currently is either *TSL* or *HKL*
Max X Points	H5T_NATIVE_INT64	The maximum number of X points in the sample grid
Max Y Points	H5T_NATIVE_INT64	The maximum number of Y points in the sample grid
SampleTransformationAngle	H5T_NATIVE_FLOAT	
SampleTransformationAngle	H5T_NATIVE_FLOAT	
Stacking Order	H5T_NATIVE_UINT32	Defines which slice corresponds to the Z=0 in the coordinate system. Optional Attribute of type H5T_STRING, with Name "Name" and Value ("Low To High" or "High To Low")
X Resolution	H5T_NATIVE_FLOAT	Resolution between sample points in the X direction
Y Resolution	H5T_NATIVE_FLOAT	Resolution between sample points in the Y direction
Z Resolution	H5T_NATIVE_FLOAT	Resolution between slices in the Z Direction
ZEndIndex	H5T_NATIVE_INT64	Starting Slice index
ZStartIndex	H5T_NATIVE_INT64	Ending Slice index (inclusive)
Slice Data organized by Slice index		

Slice Group Specification

Each Slice is grouped into its own H5G_GROUP with the Name of the group simply the index of the slice. Within each slice group are 2 more groups with names **Data** and **Header.**

Table B-2. Slice Group Specification

Name	HDF5 Type	Value
Data	H5G_GROUP	Contains all the data columns
Header	H5G_GROUP	Contains all the header entries

TSL Specification

This section details the data to be imported from a .ang file into the .h5ebsd file.

Table B-3. TSL Data Group Specification

Name	HDF5 Type	Value
Phi1	H5T_NATIVE_FLOAT	Contains all the Phi1 data in a 1D Array with length equal to the total number of points.
Phi	H5T_NATIVE_FLOAT	Contains all the Phi data in a 1D Array with length equal to the total number of points.
Phi2	H5T_NATIVE_FLOAT	Contains all the Phi2 data in a 1D Array with length equal to the total number of points.
X Position	H5T_NATIVE_FLOAT	Contains all the X position data in a 1D Array with length equal to the total number of points.
Y Position	H5T_NATIVE_FLOAT	Contains all the Y position data in a 1D Array with length equal to the total number of points.
Image Quality	H5T_NATIVE_FLOAT	Contains all the Image Quality data in a 1D Array with length equal to the total number of points.
Confidence Index	H5T_NATIVE_FLOAT	Contains all the Confidence Index data in a 1D Array with length equal to the total number of points.
PhaseData	H5T_NATIVE_INT32	Contains all the PhaseData data in a 1D Array with length equal to the total number of points.
SEM Signal	H5T_NATIVE_FLOAT	Contains all the SEM Signal data in a 1D Array with length equal to the total number of points.
Fit	H5T_NATIVE_FLOAT	Contains all the Fit of Solution data in a 1D Array with length equal to the total number of points.

Table B-4. TSL Header Group Specification

Name	HDF5 Type	Value
OriginalFile	H5T_STRING	Path to the .ang file that was imported
OriginalHeader	H5T_STRING	Contains the original header from the imported .ctf file
TEM_PIXperUM	H5T_NATIVE_FLOAT	Contains value for the header entry TEM_PIXperUM
x-star	H5T_NATIVE_FLOAT	Contains value for the header entry x-star
y-star	H5T_NATIVE_FLOAT	Contains value for the header entry y-star
z-star	H5T_NATIVE_FLOAT	Contains value for the header entry z-star
WorkingDistance	H5T_NATIVE_FLOAT	Contains value for the header entry WorkingDistance
ElasticConstants	H5T_STRING	Contains value for the header entry ElasticConstants
GRID	H5T_STRING	Contains value for the header entry GRID
XSTEP	H5T_NATIVE_FLOAT	Contains value for the header entry XSTEP
YSTEP	H5T_NATIVE_FLOAT	Contains value for the header entry YSTEP
NCOLS_ODD	H5T_NATIVE_INT32	Contains value for the header entry NCOLS_ODD
NCOLS_EVEN	H5T_NATIVE_INT32	Contains value for the header entry NCOLS_EVEN
NROWS	H5T_NATIVE_INT32	Contains value for the header entry NROWS
OPERATOR	H5T_STRING	Contains value for the header entry OPERATOR
SAMPLEID	H5T_STRING	Contains value for the header entry SAMPLEID
SCANID	H5T_STRING	Contains value for the header entry SCANID
Phases	H5G_GROUP	Group that contains a subgroup for each phase where the name of each subgroup is simply the index of the phase starting at 1.

Table B-5. TSL Phase Group Specification

Name	HDF5 Type	Value
Categories	H5T_NATIVE_INT32	
Formula	H5T_STRING	
Info	H5T_STRING	
LatticeConstants	H5T_NATIVE_FLOAT	
LatticeConstants	H5T_STRING	
NumberFamilies	H5T_NATIVE_INT32	
Phase	H5T_NATIVE_INT32	
Symmetry	H5T_NATIVE_INT32	
hklFamilies	H5G_GROUP	Contains all the HKL Family information where the number of datasets contained in this group is the number of HKL Families

Table B-6. TSL HKL Family Group Specification

Name	HDF5 Type	Value
{Based on Index of the family}. If there are 4 families then there are 4 data sets with names "0", "1", "2" and "3".	Custom: See Below	

TSL HKL C Structure for Reading Binary HKL Family

```
typedef struct
{
  int h;@n
  int k;@n
  int l;@n
  int s1;@n
  float diffractionIntensity;@n
  int s2;@n
} HKLFamily_t;
```

HKL Specification

This section details the data to be imported from a .ctf file into the .h5ebsd file.

Table B-7. HKL (.ctf) Data Group Specification

Name	HDF5 Type	Value
Phase	H5T_NATIVE_INT32	
X	H5T_NATIVE_FLOAT	
Y	H5T_NATIVE_FLOAT	
Z	H5T_NATIVE_FLOAT	Note that this ONLY appears in a "3D" .ctf data file
Bands	H5T_NATIVE_INT32	
Error	H5T_NATIVE_INT32	
Euler1	H5T_NATIVE_FLOAT	In 2D files these are in Degrees. In 3D files these are in Radians
Euler2	H5T_NATIVE_FLOAT	In 2D files these are in Degrees. In 3D files these are in Radians
Euler3	H5T_NATIVE_FLOAT	In 2D files these are in Degrees. In 3D files these are in Radians
MAD	H5T_NATIVE_FLOAT	
BD	H5T_NATIVE_INT32	
BS	H5T_NATIVE_INT32	
GrainIndex	H5T_NATIVE_INT32	
GrainRandomColourR	H5T_NATIVE_UINT8	
GrainRandomColourG	H5T_NATIVE_UINT8	
GrainRandomColourB	H5T_NATIVE_UINT8	

Table B-8. HKL (.ctf) Header Group Specification

Name	HDF5 Type	Value
OriginalFile	H5T_STRING	Path to the .ctf file that was imported
OriginalHeader	H5T_STRING	Contains the original header from the imported .ctf file
Prj	H5T_STRING	Contains value for the header entry Prj
Author	H5T_STRING	Contains value for the header entry Author
JobMode	H5T_STRING	Contains value for the header entry JobMode
XCells	H5T_NATIVE_INT32	Contains value for the header entry XCells
YCells	H5T_NATIVE_INT32	Contains value for the header entry YCells
XStep	H5T_NATIVE_FLOAT	Contains value for the header entry XStep
YStep	H5T_NATIVE_FLOAT	Contains value for the header entry YStep
ZStep	H5T_NATIVE_FLOAT	Contains value for the header entry ZStep
ZCells	H5T_NATIVE_FLOAT	Contains value for the header entry ZCells
AcqE1	H5T_NATIVE_FLOAT	Contains value for the header entry AcqE1
AcqE2	H5T_NATIVE_FLOAT	Contains value for the header entry AcqE2
AcqE3	H5T_NATIVE_FLOAT	Contains value for the header entry AcqE3
Mag	H5T_NATIVE_INT32	Contains value for the header entry Mag
Coverage	H5T_NATIVE_INT32	Contains value for the header entry Coverage
Device	H5T_NATIVE_INT32	Contains value for the header entry Device
KV	H5T_NATIVE_INT32	Contains value for the header entry KV
TiltAngle	H5T_NATIVE_FLOAT	Contains value for the header entry TiltAngle
TiltAxis	H5T_NATIVE_FLOAT	Contains value for the header entry TiltAxis
Phases	H5G_GROUP	Group that contains a subgroup for each phase where the name of each subgroup is simply the index of the phase starting at 1.

Table B-9. HKL (.ctf) Phase Group Specification

Name	HDF5 Type	Value
Comment	H5T_STRING	Contains value for the header entry Comment
Internal1	H5T_STRING	Contains value for the header entry Internal1
Internal2	H5T_STRING	Contains value for the header entry Internal2
LatticeAngles	H5T_NATIVE_FLOAT	Contains value for the header entry Lattice Angles in a 1x3 array
LatticeDimensions	H5T_NATIVE_FLOAT	Contains value for the header entry Lattice Dimensions in a 1x3 array
LaueGroup	H5T_NATIVE_INT32	Contains value for the header entry LaueGroup plus an H5T_STRING Attribute which is the string name of the Laue Group for example "Hexagonal-High 6/mmm"
SpaceGroup	H5T_NATIVE_INT32	Contains value for the header entry SpaceGroup
PhaseName	H5T_STRING	Contains value for the header entry PhaseName

Stacking Order Discussion

The **Stacking Order** refers to the order in which the z slices are stacked together when they are read from the file. The enumerations are also in the *EbsdLibConstants.h* header file.

As a further explanation if the ordering is **Low To High** then the slice with the lowest number is positioned at Z=0 in 3D Cartesian space. For example if your data set is numbered from 23 to 86 with file names of the form Slice_023.ang and you select "Low To High" then the data inside of file Slice_023.ang will be positioned at Z=0 during any method that has to deal with the data. The opposite of this is if the user were to select to have their data **High to Low** in which case the file with name Slice_086.ang will be positioned at Z=0 and the file with name "Slice_023.ang" will be positioned at Z=64.

APPENDIX C: STATSGENERATOR APPLICATION

In order to allow the user to quickly generate a set of statistics that can be used to create a synthetic microstructure the "StatsGenerator" program was created. Using this program the user can generate the necessary statistics that describe a microstructure and save those statistics to a DREAM3D file that can then be used in a DREAM3D pipeline to generate a synthetic microstructure.

The main features are:

- Describe the grain size distribution via a simple Gaussian distribution
- Select from several preset microstructure types
- Define the Omega 3 Distributions
- Define the Shape distributions
- Define the Neighbor distributions
- Define a Crystallographic Texture through ODF parameters
- Define the MDF through various parameters
- Define the Axis ODF for the grains
- Add as many phases as the user needs
- Define each phase according to several presets
    - Primary
    - Precipitate
    - Transformation
    - Matrix
    - Boundary

Size Distribution Tab

Mu: This is the average value of the lognormal grain size distribution

Sigma: This is the standard deviation of the lognormal grain size distribution

Sigma Cut Off Value: This allows the user to truncate the distribution to remove very large grains

Bin Step Size: This is the size of bin to use in segregating the grains into size classes for correlating other statistics to grain size. Note that the Bins to be Created is displayed in the bottom left corner.

Preset Statistic Models: This allows the user to select a *morphological-type* of microstructure to populate the default data.

Default: this populates the statistic tabs with generic random data that may not create a buildable microstructure. This option should only be used if the user is going to enter the values on all the statistics tabs themselves.

Equiaxed: this populates the statistic tabs with data that is designed to generate a random equiaxed microstructure.

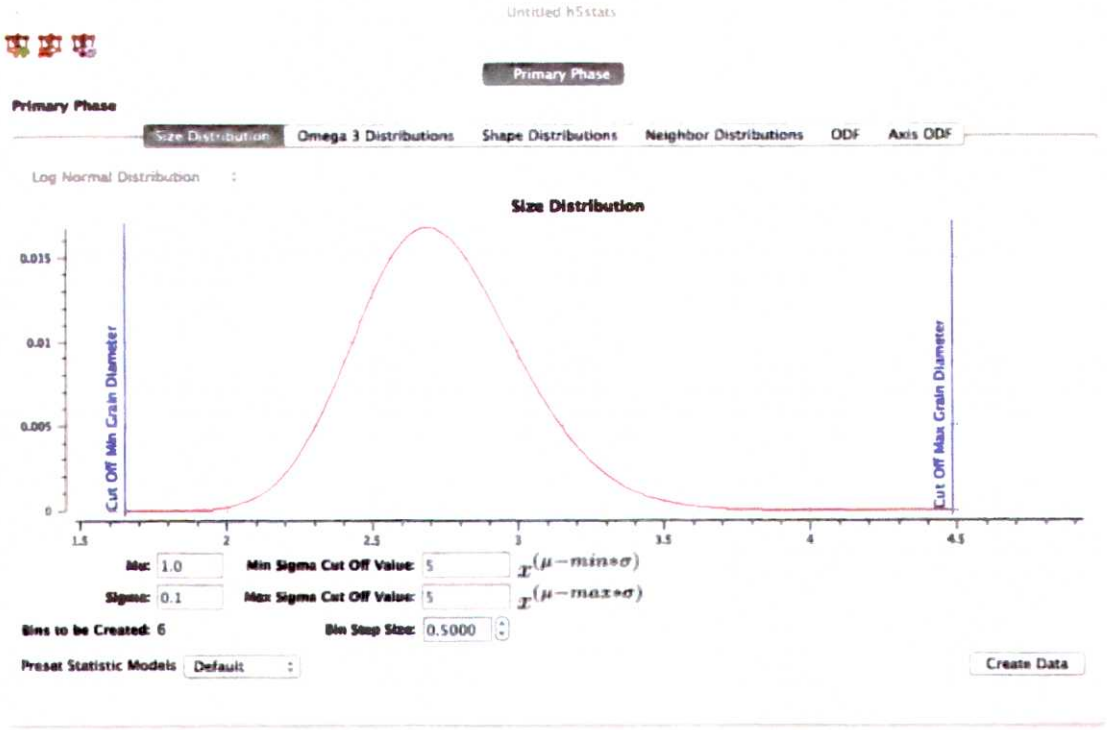

Figure C-1. StatsGenerator Main Window

Rolled: this populates the statistic tabs with data that is designed to generate a rolled microstructure with elongated grains with user defined aspect ratios.

Aspect Ratio 1: this is the aspect ratio between the dimensions of the grains in the rolling direction and transverse direction, respectively.

Aspect Ratio 2: this is the aspect ratio between the dimensions of the grains in the rolling direction and normal direction, respectively. This value must be larger than Aspect Ratio 1 or the default values will be wrong.

Phase Properties: This is the area where the user can enter information about the phase for which statistics are currently being generated. The Plus Button allows the user to add a phase, the Minus Button allows the user to remove a phase and the Wheel Button allows the user to edit the currently selected phase.

Select Crystal Structure: this allows the user to specify the crystal structure of the phase. Currently, the two options are Cubic and Hexagonal.

Fraction: this is the volume fraction of the phase. The Calculated Phase Fraction is updated as more phases are added, by scaling the current total of all the phases' fractions to 1 (in case the user's total is not equal to 1 when finished).

Select Phase Type: this specifies the type of the phase. Currently, the two types of phases are Primary and Precipitate. Note that there must be at least one Primary phase before any Precipitate phase can be created or the Synthetic Builder will fail.

Fraction of Precipitate on Boundary: if the phase type is set to Precipitate, then the user must specify the number fraction of the precipitates that are located on grain boundaries of the primary phase. This value will be scaled to 1 if the user's value is larger than 1. The value is keyed to -1 for Primary phases.

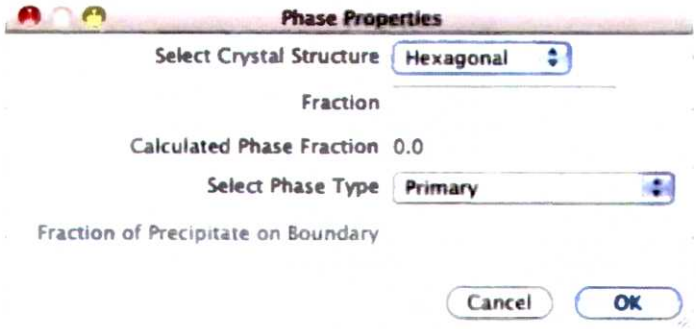

Figure C-2. Phase Properties

Create Data: This locks in the values the user has entered and populates the other tabs with default values. The user can move through the other tabs and change any default values if desired.

Omega3 Tab

Bin: this column is calculated from the size distribution and cannot be changed.

Alpha: this is the alpha parameter of a Beta distribution. Omega 3 is normalized and can only be between 0 and 1, so the Beta distribution is a good fit

Beta: this is the beta parameter of the Beta distribution.

Color: this allows the user to change colors of the curves for image creation or easier identification during stats generation.

Shape Distribution Tab

Bin: this column is calculated from the size distribution and cannot be changed.

Alpha: this is the alpha parameter of a Beta distribution. B/A, C/A and C/B are normalized and can only be between 0 and 1, so the Beta distribution is a good fit.

Beta: this is the beta parameter of the Beta distribution.

Color: this allows the user to change colors of the curves for image creation or easier identification during stats generation.

Neighbor Distribution Tab

- **Bin**: this column is calculated from the size distribution and cannot be changed.
- **Alpha**: this is the alpha parameter of a Power Law distribution.
  - this is the exponent of a Power Law Distribution.
- **Beta**: this is the beta parameter of a Power Law distribution.
- **Color**: this allows the user to change colors of the curves for image creation or easier identification during stats generation.

ODF Tab

Weights and Spreads Sub-Tab

- **Euler 1-3:** these are the Euler angles that define an orientation that the user would like to increase in weight.
- **Weight:** this is the weight in MRD (multiples of random).
- **Sigma:** this is the spread to use in blurring out the orientation chosen. The value corresponds to the number of bins in Rodrigues (orientation) space it takes for the MRD value entered in the Weight column to reduce to 0.0 (decreasing quadratically from the bin of the entered orientation).
- **Calculate ODF:** this builds the ODF and then creates pole figures (PFs) for the user to inspect.

Pole Figure (PF) Sub-Tabs

There are three PFs formed for each of the crystal structures that can be chosen (though they are of different directions for the different crystal structures).

MDF Sub Tab: This sub-tab will display the baseline MDF for the generated ODF. The implemented algorithm proceeds by randomly sampling pairs of orientations from the ODF and calculating the misorientation (axis-angle only). Only the angle is plotted in the misorientation distribution plot. The user can also add axis-angle pairs to increase in weight.

- **Angle:** this is the angle of the misorientation to increase in weight.
- **Axis:** this is the axis of the misorientation to increase in weight. If the crystal structure being used for the phase is Hexagonal, then this axis is in the 3-index, orthogonal convention, not the true (hkil) convention.
- **Weight:** this is the weight in units of MRD (multiples of random) of the entered misorientation.

Axis ODF Tab

Weights and Spreads Sub-Tab

- **Euler 1-3:** these are the Euler angles that define an orientation that the user would like to increase in weight.
- **Weight:** this is the weight in MRD (multiples of random) to be assigned to the orientation listed.
- **Sigma:** this is the spread to use in blurring out the orientation chosen. The value corresponds to the number of bins in Rodrigues (orientation) space it takes for the MRD value entered in the Weight column to reduce to 0.0 (decreasing quadratically from the bin of the entered orientation).

- **Calculate ODF:** this builds the ODF and then creates pole figures (PFs) for the user to inspect.

Pole Figure (PF) Sub-Tabs

There are three pole figures formed, which correspond to the location of the 3 principal axes of the grains to be generated (i.e a > b > c).

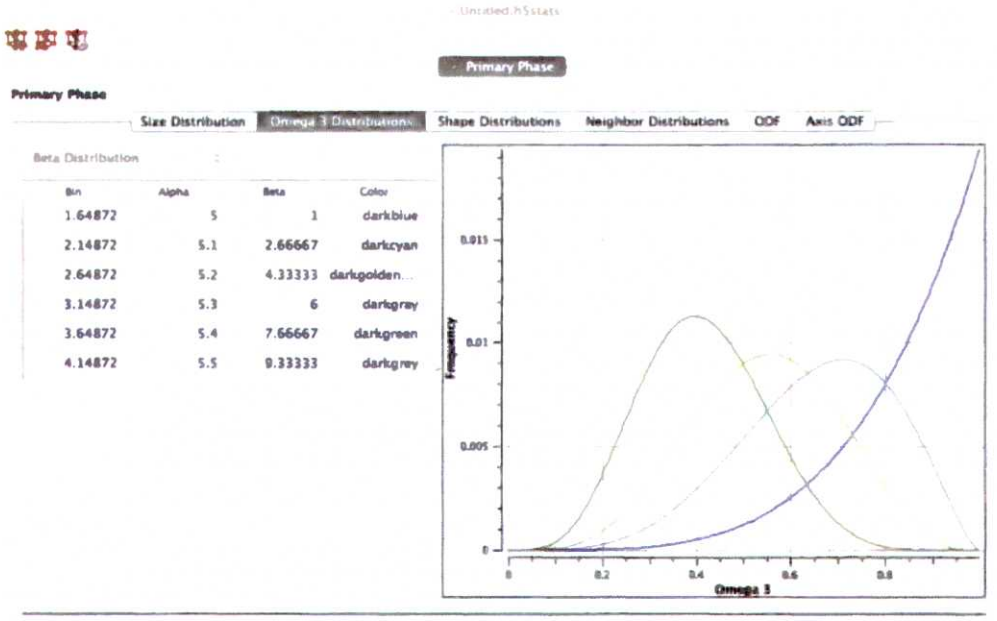

Figure C-3. Omega 3 GUI

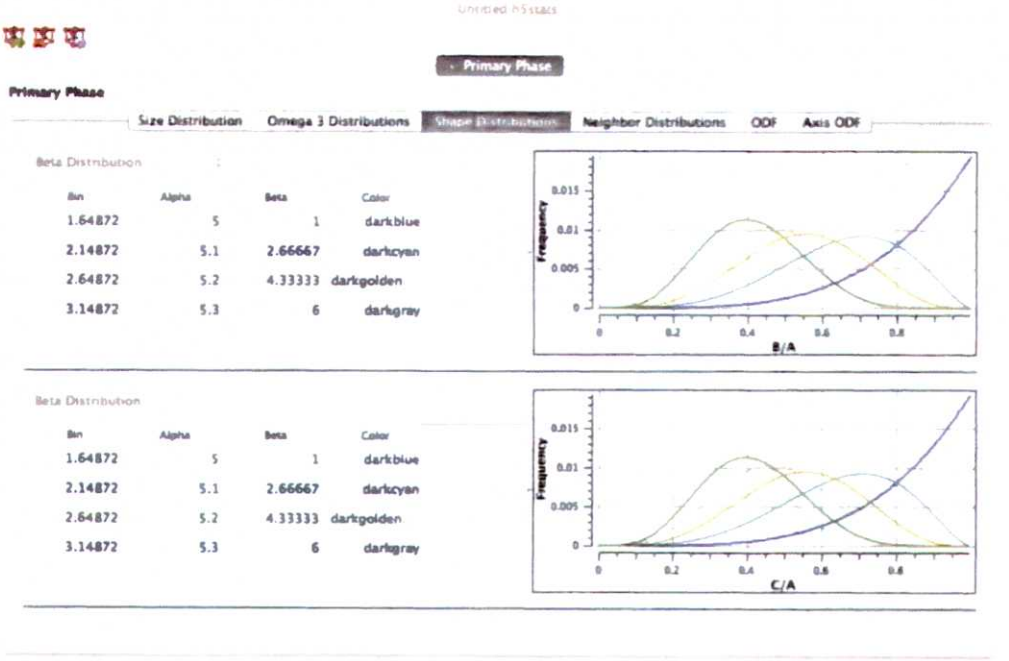

Figure C-4. Shape Distribution Function GUI

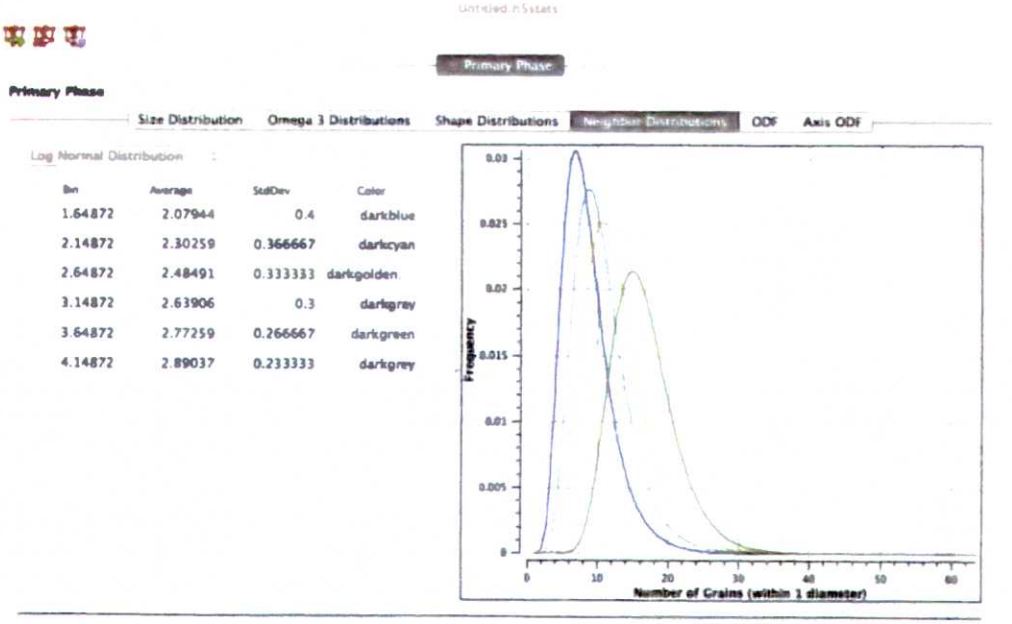

Figure C-5. Neighbor Distribution Function GUI

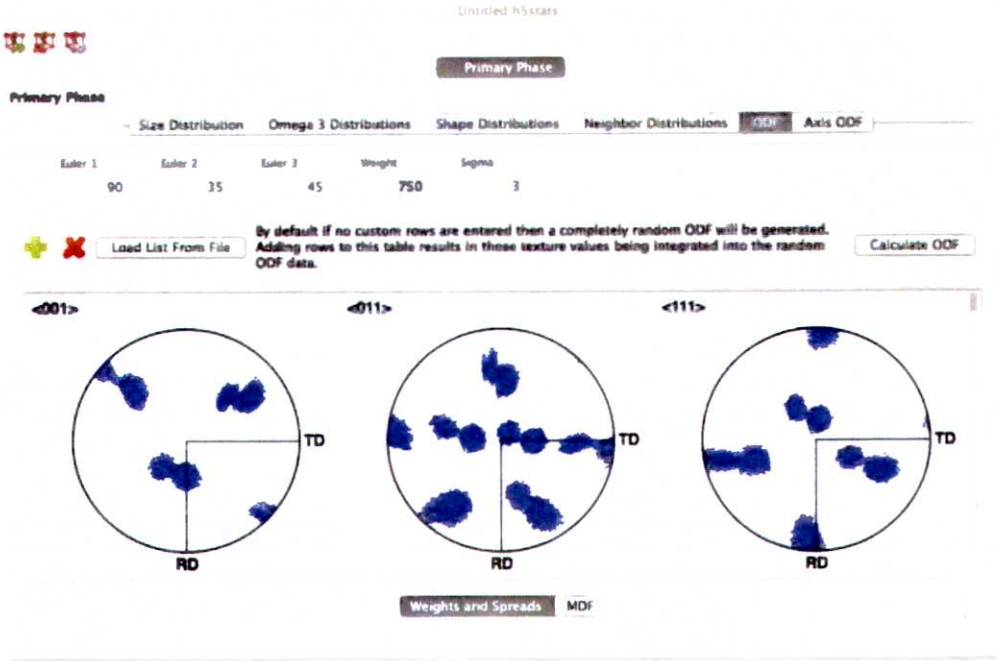

Figure C-6. ODF GUI

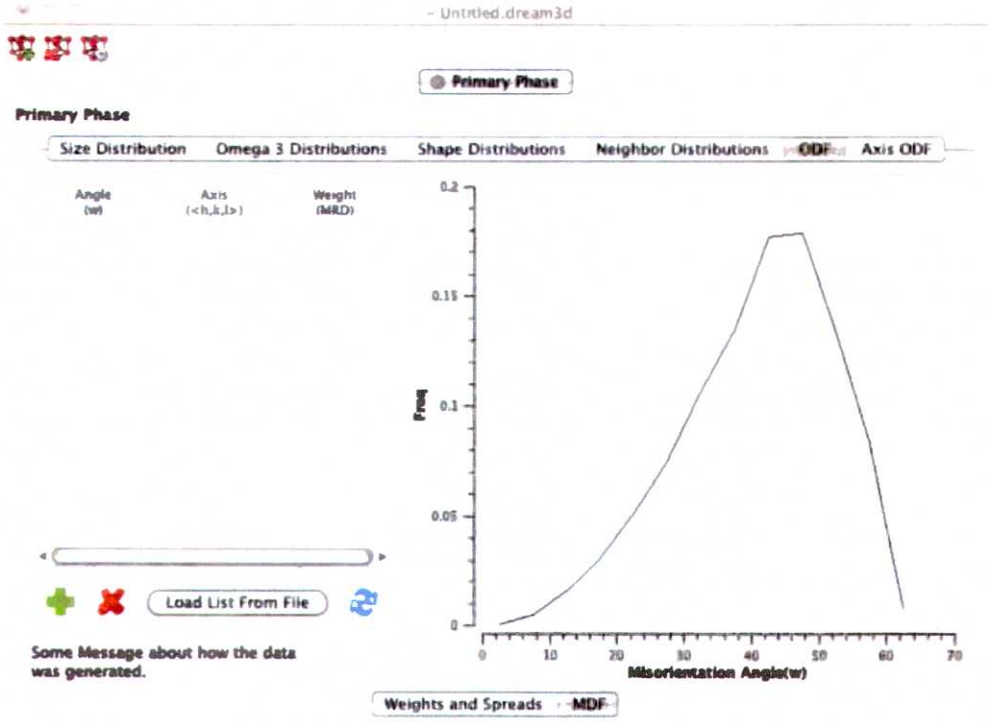

Figure C-7. MDF GUI

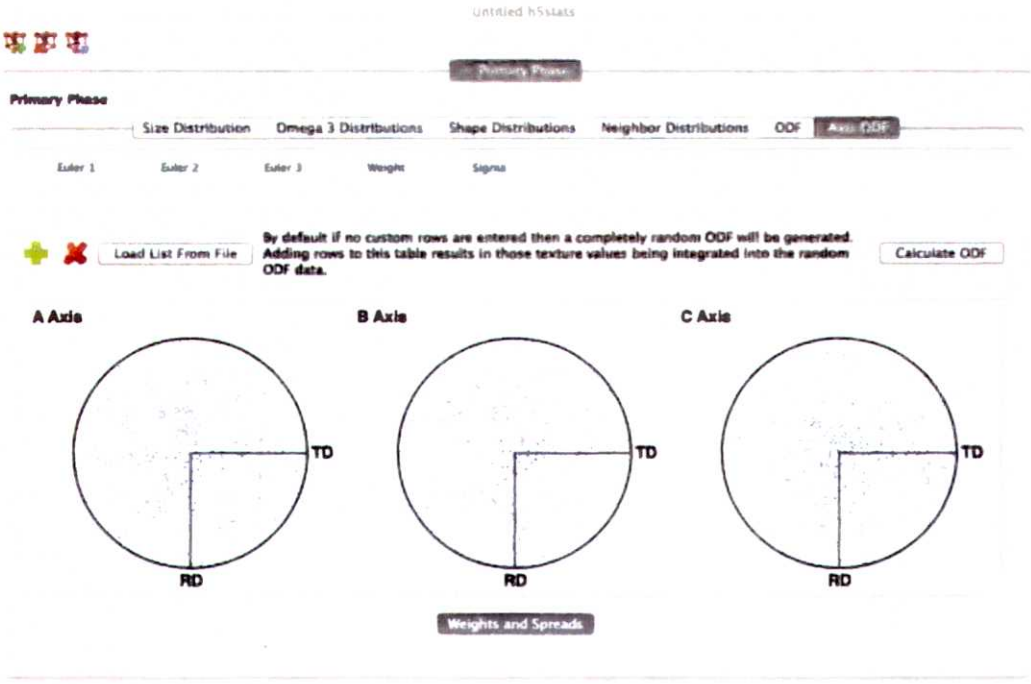

Figure C-8. Axis ODF GUI

LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS

Acronym	Description
AFRL	Air Force Research Laboratory
BSE	BackScatter Electron Image
DOD	Department of Defense
DREAM3D	Digital Representation Environment for Analysis of Microstructure
DTIC	Defense Technical Information Center
EAR	Export Administration Regulation
EBSD	Electron BackScatter Diffraction
EDS	Energy Dispersive X-ray Diffraction
FEM	Finite Element Methods
GUI	Graphical User Interface
HPC	High Performance Computing
IISE	Ion Induced Secondary Electron
ICME	Integrated Computational Materials Science
ITK	Image Processing Toolkit
ITAR	International Traffic in Arms Regulation
MDF	Misorientation Distribution Function
ODF	Orientation Distribution Function
OpenCV	Open Computer Vision
PF	Pole Figure
RTF	Rich Text Format
RX	Materials and Manufacturing Directorate
SEI	Secondary Electron Image
SNR	Signal to Noise Ratio
SSH	Secure Shell
TBB	Thread Building Blocks
USAF	United States Air Force
WPAFB	Wright-Patterson Air Force Base